2013 IEEE 13th International Conference on Data Mining

Power to the Points: Validating Data Memberships in Clusterings

Parasaran Raman
School of Computing, University of Utah
Salt Lake City, Utah
praman@cs.utah.edu

Suresh Venkatasubramanian
School of Computing, University of Utah
Salt Lake City, Utah
suresh@cs.utah.edu

***Abstract*—In this paper, we present a method to attach affinity scores to the implicit labels of individual points in a clustering. The affinity scores capture the confidence level of the cluster that claims to "own" the point. We demonstrate that these scores accurately capture the quality of the label assigned to the point. We also show further applications of these scores to estimate *global* measures of clustering quality, as well as accelerate clustering algorithms by orders of magnitude using active selection based on affinity.**

This method is very general and applies to clusterings derived from any geometric source. It lends itself to easy visualization and can prove useful as part of an interactive visual analytics framework. It is also efficient: assigning an affinity score to a point depends only polynomially on the number of clusters *and is independent both of the size and dimensionality of the data*. It is based on techniques from the theory of interpolation, coupled with sampling and estimation algorithms from high dimensional computational geometry.

***Keywords*-Natural Neighbor Interpolation; Validating Clusterings; Power Diagrams;**

I. Introduction

Clustering is an unsupervised exploratory data mining technique that generates predictions in the form of implicit labels for points. These predictions are used for exploration, data compression, and other forms of downstream data analysis, and so it is important to verify the accuracy of these labels. However, because of the unsupervised nature of clustering, there is no direct way to validate the data assignments. As a consequence, a number of indirect approaches have been developed to validate a clustering at a *global* level[1, 2]. These include internal, external and relative validation techniques, and methods based on *clustering stability* that assume a clustering (algorithm) is good if small perturbations in the input do not affect the output clustering significantly[1].

But all these approaches are global. They assign a *single number* to a clustering and cannot capture the potentially wide variation in label quality within a clustering. Consider for example a clustering of the MNIST `digits` database with a few example images displayed in Figure 1.

Figure 1. MNIST Handwritten digits. L-R are numbers {0,6,4,9}. The numbers on the top row are very hard to identify even for a human. The bottom row is unambiguous.

By global measures of clusterability, the clustering would be considered "good". However, as we can see in the picture in the top row, there are a number of images for which the correct cluster is not as obvious. What we would like in this case is a way to quantify this lack of confidence for *each image* separately. Such a measure would give a lower confidence rating to the labels for images in the top row, and a downstream analysis task could incorporate this uncertainty into its reasoning. A single number describing the quality of the clustering would not suffice in this case, because the downstream analysis might only select a few points (cluster centers, or a representative sample) for further processing.

A. Our Work

In this paper we present a scheme to assign local affinity scores to points that indicate the "strength" of their assignment to a cluster. Our approach has a number of attractive features.

- it is very general: it takes a clustering generated by any method and returns the local affinity scores without relying on probabilistic or other modeling assumptions. It does this by using the ideas of proximity and shared volume: intuitively, *a point has strong affinity for a cluster if (when treated as a singleton cluster) its region of influence overlaps significantly with the region of influence of the cluster.*
- it is very efficient to compute: computing the local affinity of a point depends solely on the number of clusters in the data and an error parameter: *there is*

This research was supported by NSF award CCF-0953066.

[1]There are supervised variants of clustering. However, these typically require domain knowledge, and the immense popularity of clustering comes precisely from the fact that it can be applied as a first filter to acquire a deeper understanding of the data.

1550-4786/13 $31.00 © 2013 IEEE
DOI 10.1109/ICDM.2013.147

no dependence on the data size or dimensionality. We show that this can be improved further by progressive refinement, allowing us to avoid computing affinities for points that we are very confident about.
- it lends itself to easy visualization, which is very useful for diagnostic purposes.
- the local affinities we compute can also be used to validate the number of clusters in the data as well speeding up clustering computations by focusing attention on points that can affect decision boundaries (as with active learning techniques).

B. Overview of our ideas

Clustering is about *proximity*: points are expected to have similar labels if they are close to each other and not to others. In other words, the *regions of influence* of points belonging to the same cluster must overlap [3]. Therefore, a point should be associated with a cluster if its region of influence significantly overlaps the region of influence of the cluster, and does not have such an overlap with other clusters. And more importantly, we can quantify the confidence of this association by measuring the *degree* of overlap.

The method we propose elaborates on this idea to incorporate a variety of more general notions of regions of influence that can incorporate cluster importance, density and even different cluster shapes. The key idea is to define regions of influence as elements of an appropriate weighted power diagram (a generalization of a Voronoi diagram) and use shared volume to quantify how different regions overlap.

At first glance, this idea is doomed to fail: computing Voronoi regions (and their volumes) is extremely difficult in high dimensions. We show how the volumes of these regions can be estimated (a) without actually computing them and (b) with provable guarantees on the estimates via the use of ϵ-net-based sampling and techniques for sampling from convex bodies in high dimensions efficiently. The resulting scheme is accurate and yields the affinity score of a point in time independent of the data size and dimensionality. It runs extremely fast in practice, taking only milliseconds to compute the scores. These scores can also be computed progressively using iterative refinement, so we can focus on the problem cases (points of low affinity) directly.

C. Applications

The local affinity scores we compute can be viewed as a general diagnostic tool for evaluating clusterings and even computing clusterings faster. We demonstrate this with a set of key applications.

Evaluating the clusterability of data. We have already explained how we expect local affinity scores to certify whether data labels are accurate or not. In addition, combining local affinity scores provides another measure for the global quality of a clustering. We will show that this measure matches prior notions[2, 4] of global quality of a clustering and thus is a more general tool for clustering quality. We will also show that this global measure can be used to solve the vexing problem of identifying the right number of clusters in a clustering [5, 6, 7], and has certain advantages over other approaches like the often-used "elbow method"[7] which looks at the point marginal gain of adding an additional cluster drops.

Active Clustering. Clustering algorithms usually have a non-linear time dependence on the input size, and so as data sizes grow, the time to cluster grows even faster. This motivates "bootstrapping" strategies where the algorithm first clusters a small sample of the data, and uses this partial clustering to find points that lie on cluster boundaries (and would have greater influence on the resulting clustering). The most important step in this "active" approach to clustering[8, 9, 10] is selecting the points to add to the process. We show that if we use points of low affinity as the active points used to seed the next round of clustering, we can obtain accuracy equal to that obtained from the entire data set but with orders of magnitude faster running time.

II. Background

Clusterings can be validated globally in three different ways [1]. *Internal* validation mechanisms look at the structure of a clustering and attempt to determine its quality[4]. For example, the ratio of the minimum inter-cluster distance to the maximum intra-cluster distance is a measure of how well-separated clusters are, and thus how good the clustering is. *External* validation measures can be employed when a reference clustering exists. In this case, an appropriate distance between clusterings must be defined, and then the given clustering can be compared to the reference clustering[2]. *Relative* validation measures look at different runs of a clustering algorithm and compare the resulting clusterings produced[2].

Cluster stability[11, 12, 13] is another way to validate clusterings. The goal here is to determine how robust a clustering solution is to small perturbations in algorithm parameters. This idea was used to do model selection; for example, the "right" number of clusters is the one that exhibits the most stable clusterings. Stability in general has been studied extensively in the statistics and machine learning communities, as a way to understand generalization properties of algorithms. The paper by Elisseeff et al. [14] provides a good overview of this literature and the monograph by Luxburg[15] focuses on clustering.

Probabilistic Modeling: Where admissible (for example when effective models of the data can be built), probabilistic modeling yields posterior likelihoods for a cluster assignment in the form of conditional probabilities $p(C \mid \mathbf{x})$ for point $\mathbf{x}$ and cluster C. We view our approach as complementary to (and more general than) model-based validation. Our approach is purely data-driven with no further assumptions, which is appropriate when initially exploring a data set.

We also show that the affinity scores produced by our method closely match the likelihoods produced by a standard clustering approach like GMMs. Note that probabilistic modeling can be used to choose a *particular* way of clustering the data, but in the setting we consider, a clustering is already given to us (possibly even by consensus clustering or some other method), and the goal is to validate it.

Validation versus outlier detection: Local validation bears a superficial resemblance to outlier detection: in both cases the goal is to evaluate individual points based on how well they "fit" into a clustering. There are important differences though. An outlier affects the *cost* of a clustering by being far away from any cluster, but it will usually be clear what cluster it might be assigned to. In contrast, a point whose *labeling* might be invalid is usually in the midst of the data. Assigning it to one cluster or another might not actually change the clustering cost, even though the label itself is now unreliable.

III. Preliminaries

Let P be a set of n points in $\mathbb{R}^d$. We assume a distance measure D on $\mathbb{R}^d$, which for now we will take to be the Euclidean distance. A *clustering* is a partition of P into clusters $C = \{C_1, C_2, \ldots, C_k\}$. We will assume that we can associate a representative $\boldsymbol{c_i}$ with a cluster C_i. For example, the representative could be the cluster centroid, or the median.

A *Voronoi diagram* [16] on a set of sites $S = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k\} \subset \mathbb{R}^d$ is a partition of $\mathbb{R}^d$ into regions $V_1, \ldots V_k$ such that for all points in V_i, the site s_i is the closest neighbor. Formally, $V_i = \{\mathbf{p} \in \mathbb{R}^d \mid D(\mathbf{p}, \mathbf{s}_i) \leq D(\mathbf{p}, \mathbf{s}_j), j \neq i\}$. When D is the Euclidean distance, the boundary between two regions is always a hyperplane, and therefore each cell V_i is a convex polyhedron with at most $k-1$ faces.

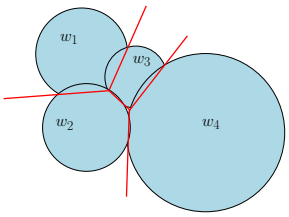


Figure 2. The power diagram of a set of points. The sphere radius is proportional to the weight w

We will also make use of a generalization of the Voronoi diagram called the *power diagram* [17]. Suppose that we associate an importance score w_i with each site $\mathbf{s}_i$. Then the power diagram on S (see Figure 2) is also a partition of $\mathbb{R}_d$ into k regions V_i, such that $V_i = \{\mathbf{p} \in \mathbb{R}^d \mid D^2(p, \mathbf{s}_i) - w_i \leq D^2(\mathbf{p}, \mathbf{s}_j) - w_j, j \neq i\}$. Power diagrams allow different sites to have different influence, but retain the property that all boundaries between regions are hyperplanes and all regions are polyhedra in Euclidean space[2].

Finally, we will frequently refer to the *volume* $\text{Vol}(S)$ of a region $S \subset \mathbb{R}^d$. In general, this denotes the d-dimensional volume of S with respect to the standard Lebesgue measure on $\mathbb{R}^d$. If S is not full-dimensional, this should be understood as referring to the lower-dimensional volume, or the volume of the relative interior of S; for example the "volume" of a triangle in three dimensions is its area, and the volume of a line segment is its length.

[2]The squared distance is crucial to making this happen; without it, arcs could be elliptical or hyperbolic.

IV. Defining Affinity Scores

As we discussed in Section I, the *region of influence* of a point is how we define its affinity to clusters. Each cluster has a region of influence. If we now consider a particular point in the data and treat it as a singleton cluster, its region of influence will overlap neighboring clusters. We measure the affinity of a point to a cluster to be the *proportion of influence it overlaps from that cluster*. We now define these ideas formally.

Defn 4.1 (Region of Influence): Let $\mathcal{C} = C_1, C_2, \ldots C_k$ be a clustering of n points. A *region of influence function* is a function $R : \mathcal{C} \to 2^{\mathbb{R}^d}$ on $\mathcal{C}$ such that all $R(C_i)$ (which are subsets of $\mathbb{R}^d$) are disjoint.

The simplest region of influence function is a Voronoi cell. Specifically, consider a clustering with k clusters, each cluster C_i having representative $\boldsymbol{c_i}$. Let C be the set of these representatives. Consider any point $x \in CH(C)$ (the convex hull of C). Let $V_1, V_2, \ldots, V_k$ be the Voronoi partition of C, and let $U_1, U_2, \ldots, U_k, U_x$ be the Voronoi partition of $C \cup \{x\}$, with U_x being the Voronoi cell of x. Then we define the region of influence, $R(C_i) = V_i$, and $R_x(C_i) = U_i$.

Defn 4.2 (Affinity Scores): Let R be a region-of-influence function. Let $\mathcal{C} = C_1, C_2, \ldots C_k$ be a clustering. For any point x, let $\mathcal{C}_x$ denote the clustering $C_1 \setminus \{x\}, C_2 \setminus \{x\}, \ldots, C_k \setminus \{x\}, \{x\}$, and let $R_x(C)$ denote the region of influence of a cluster $C \in \mathcal{C}_x$. Then the affinity score of x is the vector $(\alpha_1, \alpha_2, \ldots, \alpha_k)$, where

$$\alpha_i = \frac{\text{Vol}(R(C_i) \cap R_x(\{x\}))}{\text{Vol}(R_x(\{x\}))}$$

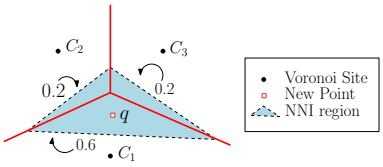


Figure 3. In this example, the red point is "stealing" the shaded area from the Voronoi cells of C_1, C_2, C_3.

In the above definition, $R_x(\{x\})$ is the region of influence x has carved out for itself, and α_i merely captures the proportion of $R_x(\{x\})$ that comes from the (original) cluster C_i. Note that all Voronoi regions can be bounded by drawing an axis aligned bounding box around the points.

Continuing our example of Voronoi-based regions of influence, the Voronoi cell U_x of x "steals" volume from Voronoi cells around it (Figure 3 illustrates this concept). We can compute the fraction of U_x that comes from any

other cell. For any point $p_i \in P$, let $\alpha_i = \frac{\text{Vol}(V_i \cap U_x)}{\text{Vol}(U_x)}$. Then α_i represents the (relative) amount of volume that x "stole" from p_i. Note that $\sum \alpha_i = 1$, and if $x = p_i$, then $\alpha_i = 1$.

The affinity score captures the entire set of interactions of a point with the clusters. It is often convenient to reduce this to a single score value. For example, since at most one α_i can be strictly greater than 0.5, we can define a point as *stable* if such an α_i exists, and say that it is assigned to cluster i. In general, we will define the *stability* of a point to be $\sigma(\mathbf{p}) = \max \alpha_i$. The stability of a point lies between 0 and 1 and a larger value indicates greater stability.

Note: The idea of *area stealing* was first defined in the context of natural neighbor interpolation[18], where the α_i values were then used to compute an interpolation of function values at the p_i. In this paper we will use the α_i directly without computing any interpolants.

A. A Rationale For Affinity

The simplest way to define influence is by distance. For example, we could define the affinity of a point to a cluster as the (normalized) distance between the point and the cluster representative. Our definition of affinity generalizes distance ratios: in one dimension, affinity calculations yield the same result as distance ratios, since the "area" stolen from a cell is merely half the distance to that cell. But affinity can capture stronger spatial effects, as our next example shows.

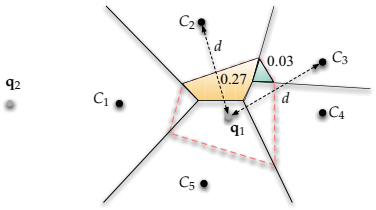


Figure 4. Illustration of the difference between distance-based and area-based influence measures

Consider the configuration shown in Figure 4. The point $\mathbf{q}_1$ is equidistant from the cluster centers $\boldsymbol{c_2}$ and $\boldsymbol{c_3}$ and so would have the same distance-based influence with respect to these clusters. But when we examine the configuration more closely, we see that the presence of $\boldsymbol{c_4}$ is reducing the influence of $\boldsymbol{c_3}$ on $\mathbf{q}_1$, and this effect appears only when we look at a *planar* region of influence. We validate using by 100 runs of k-means with random seeds. We observe that $\mathbf{q}_1$ was assigned to $\boldsymbol{c_2}$ in 15 runs and to $\boldsymbol{c_3}$ in only 2 runs. A distance-based affinity would have suggested an equal "affinity" for the two clusters, whereas a volume-based affinity incorporates the effects of other clusters.

Similarly, $\mathbf{q}_2$ is twice as close to $\boldsymbol{c_1}$ compared to $\boldsymbol{c_2}$ or $\boldsymbol{c_5}$, which would result in the distance-based influence of $\boldsymbol{c_1}$ being equal to the influence of $\boldsymbol{c_2}$ and $\boldsymbol{c_5}$ combined. When we validate this using k-means, we find that $\mathbf{q}_2$ is *exclusively* assigned to cluster center $\boldsymbol{c_1}$. Here, C_1 has a "shielding" effect on q_2 that prevents it from ever being assigned to those clusters: this shielding can only be detected with a truly spatial affinity measure.

B. Visualization

The affinity scores define a vector field over the space the data is drawn from. The stability $\sigma(\mathbf{p})$ defines a scalar field and can be visualized (in low dimensions). Consider the clustering depicted in Figure 5(a). We can draw a contour map where each level connects points with the same stability score (unlike in a topographical map, more deeply nested contours correspond to *lower* stability scores). We can also render this as a greyscale heatmap (where the lower the affinity, the brighter the color). These visualizations, while simple, provide a visual rendering of affinity scores that is useful as part of an exploratory analysis pipeline.

C. Extensions

Our definition of affinity is not limited to Euclidean spaces. It can be generalized to a variety of spaces merely by modifying the way in which we construct the Voronoi diagrams. In all cases, the resulting affinity scores will result from a volume computation over polyhedra.

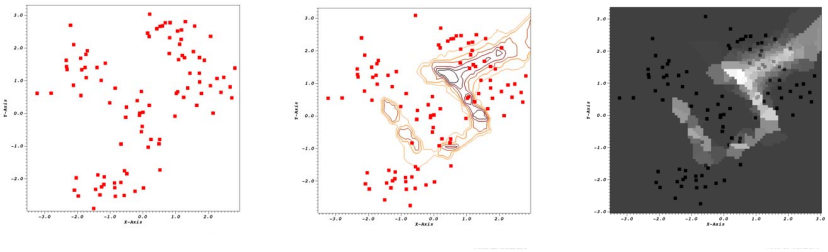
Giving clusters varying importance: density-based methods: Consider a generalized clustering instance where each cluster C_i has an associated weight w_i, with a larger w_i indicating greater importance. Instead of constructing the Voronoi diagram, we will construct the *power diagram* defined in Section III. Specifically, the region of influence R_i for cluster C_i will be defined as the set $R(C_i) = \{x | d^2(p_i, x) - w_i \leq d^2(p_j, x) - w_j\}$. We compute the affinity vector as before, with the weight of a singleton $\mathbf{x}$ set appropriately depending on the weight function used. For example, if $w(C_i) = |C_i|/n$, then $w(\mathbf{x}) = 1/n$.

Consider the examples depicted in Figure 6. The left-hand figure has 100 points in each of five clusters, and the right-hand figure has 500 points in each of four outer clusters and 100 points in the center cluster. Notice that there is a lot more instability (as seen by the contours) in the sparser example, much of which is due to the presence of the central cluster. However, once the density of the outer clusters increases, the effect of the inner cluster is much weaker, and there are fewer unstable regions.

We can also extend our Voronoi-based definition of affinity to clusterings in Bregman spaces[19] and kernel spaces[20] by reducing the resulting affinity score to volume computation on polyhedra, just as in the Euclidean space. We omit further discussion of these settings in the interest of space.

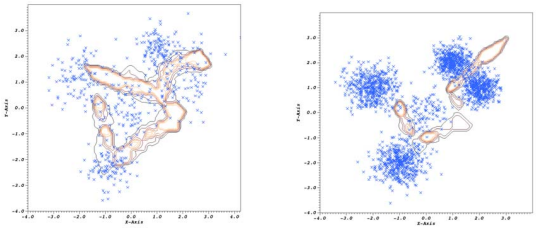
V. Estimating Affinity

The many different ways of defining affinity scores via regions of influence all reduce to the following: given a set of representatives $C = \{c_1, \ldots, c_k\}$ and a query point x, estimate the volume of a single cell in the Voronoi diagram



(a) Data in five clusters (b) A contour plot (c) Heat map

Figure 5. Visualizing the affinity scores



(a) A data set with 100 points in each cluster

(b) A data set with 100 points in the *center* cluster and 500 points in the rest.

Figure 6. Density changes affinity regions

of C or $C \cup \{x\}$, and estimate the volume of the intersection of two such cells.

In two dimensions, the Voronoi (or weighted Voronoi) diagram of k points can be computed in time $O(k \log k)$[16], and the intersection of two convex polygons can be computed in $O(k)$ time[21]. Any polygon with k vertices can be triangulated in $O(k)$ time using $O(k)$ triangles, and then the area can be computed exactly in $O(k)$ time ($O(1)$ time per triangle). In three dimensions, computing the Voronoi diagram takes $O(k^2)$ time, and computing the intersection of two convex polyhedra can be done in linear time [22]. Tetrahedralizing the convex polyhedron can also be done in linear time.[23].

This direct approach to volume computation does not scale. In general, a single cell in the Voronoi diagram of k points in $\mathbb{R}^d$ can have complexity $O(k^{\lceil d/2 \rceil})$. We now propose an alternate strategy that provably approximates the affinity scores to any desired degree of accuracy in polynomial time using random sampling.

Let $U_{\mathbf{x}}$ be the Voronoi cell of $\mathbf{x}$ in the Voronoi diagram of $C \cup \{\mathbf{x}\}$. We say that the point $\mathbf{y}$ is *stolen from* $s(\mathbf{y}) \triangleq c_i$ if (i) $\mathbf{y} \in U_{\mathbf{x}}$ and (ii) $\mathbf{y}$'s second nearest neighbor is $\mathbf{c}_i$. We can then write $\alpha_i = \frac{\text{Vol}(\{\mathbf{x} | s(\mathbf{x}) = \mathbf{c}_i\})}{\text{Vol}(U_{\mathbf{x}})}$. Note that given a point $\mathbf{x}$ and any point $\mathbf{y}$, we can verify in $O(k)$ time whether $\mathbf{y} \in U_{\mathbf{x}}$ and also compute $s(\mathbf{y})$ by direct calculation of the appropriate distance measure.

Let $(\alpha_1, \alpha_2, \ldots, \alpha_k)$ be the affinity scores for x. Suppose we now sample a point $\mathbf{y}$ uniformly at random from $U_{\mathbf{x}}$. We can find $s(\mathbf{y})$ in $O(k)$ time and this provides one update to α_i. The number of such samples needed to get an accurate estimate of each α_i is given by the theory of ε-samples. Let μ be a measure defined over X and let $\mathcal{R}$ be a collection of subsets of X. An *ε-sample* with respect to $(X, \mathcal{R})$ and μ is a subset $S \subset X$ such that for any subset $R \in \mathcal{R}$,

$$\left| \frac{\mu(S \cap R)}{\mu(S)} - \frac{\mu(R)}{\mu(X)} \right| \leq \varepsilon.$$

By standard results in VC-dimension theory[24], a random subset of size $O(\frac{d}{\varepsilon^2})$ is an ε-sample for a range space $(X, \mathcal{R})$ of VC-dimension d.

If we now consider the discrete space $[1 \ldots k]$ with the measure $\mu(i) = \alpha_i$, then the set of ranges $\mathcal{R}$ is the set of singleton queries $\{1 \ldots k\}$, and the VC-dimension of $([1 \ldots k], \mathcal{R})$ is a constant. This means that if we sample a set S of $O(\frac{d}{\varepsilon^2})$ points from $U_{\mathbf{x}}$, and set $\tilde{\alpha}_i = \frac{|\{x \in S | s(x) = i\}|}{|S|}$, then $|\tilde{\alpha}_i - \alpha_i| \leq \epsilon$ for all i.

A. *Sampling from $U_{\mathbf{x}}$*

We now have a strategy to estimate the affinity scores of $\mathbf{x}$. Sample the number of points from $U_{\mathbf{x}}$ as prescribed above and then estimate $\tilde{\alpha}_i$ by computing the owners of samples. Standard rejection sampling (sample from a ball enclosing $U_{\mathbf{x}}$ and reject points outside it) does not work in high dimensions as the number of rejected points grows exponentially with the dimension. For example in twenty dimensions, over one thousand points are rejected for each good sample in experiments.

To solve this problem, we make use of the extensive literature on sampling from a convex polyhedron in time polynomial in d, following a groundbreaking randomized polynomial time algorithm[25]. At a high level, these are all MCMC methods: they use different random walks to

Algorithm 1 SamplePolytope

Input: Collection of halfplanes $\mathcal{H}$ defining convex region $K = \cap_{h \in \mathcal{H}} h$, number of samples m.
Output: m points uniformly sampled from K.

Construct affine transform T such that TK is centered and isotropic.
Fix *burn-in* parameter b
Run **Hit-And-Run** for d steps on TK, ending in $z = z_0$
for $i = 1 \ldots m$ **do**
  Set z_i to be result of one **Hit-And-Run** move from z_{i-1}
Return $(T^{-1}z_1, \ldots, T^{-1}z_m)$.

extract a single uniform sample from the polyhedron efficiently. We describe the sampling procedure in Algorithm 1. One of the most effective strategies in practice for doing this is known as *hit-and-run*[26]. It works as follows. Starting with some point $\mathbf{x}$ in the desired polytope K, we pick a direction at random, and then pick a point uniformly on the line segment emanating from $\mathbf{x}$ in that direction and ending in the boundary of K. We refer to this step as **Hit-And-Run**. It has been shown[27] that this random walk mixes very well, making $O(d^3)$ calls to a membership oracle to produce a single sample (under some technical assumptions). Figure 7 illustrates the uniformity in the distribution of samples using **Hit-And-Run** for the Voronoi cell of the point q with just a few samples.

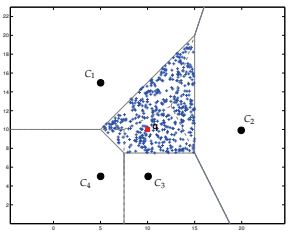


Figure 7. Illustration of **Hit-And-Run** for sampling from a Voronoi cell. Samples are shown in blue.

Algorithm 2 (AFFINITY) summarizes the process for computing the affinity score of a single point.

Reducing dimensionality: The above sampling procedure runs in time $O(d^3)$ per point. However, d can be quite large. We make one final observation that replaces terms involving d by terms involving $k \ll d$ for Euclidean distance measures (or Euclidean distances derived from a kernel).

The Voronoi diagram of k points in d dimensions, where $k < d$, has a special structure. The k points together define a $k-1$-dimensional subspace $\mathcal{H}$ of $\mathbb{R}^d$. This means that any vector $p \in \mathbb{R}^d$ can be written as $p = u + w$ where $u \in \mathcal{H}$ and $w \perp u$. The Euclidean distance $\|p - p'\|^2$ can be written as $\|u - u'\|^2 + \|w - w'\|^2$. In particular, this means that in any subspace of the form $\mathcal{H} + w$ for a fixed $w \perp \mathcal{H}$, the distance between two points is merely their distance in $\mathcal{H}$.

Therefore, each Voronoi cell V can be written as $V' + \mathcal{H}^\perp$, where $V' \subset \mathcal{H}$ and $\mathcal{H}^\perp$ is the orthogonal complement of

Algorithm 2 AFFINITY: Computing the affinity score for a point

Input: A clustering $\mathcal{C} = C_1, C_2, \ldots, C_k$ with representatives $c_1, \ldots, c_k$ and a point x.
Output: Affinity vector $(\alpha_1, \ldots, \alpha_k)$ for x

$m \leftarrow \frac{c}{\varepsilon^2}$
Set all $\alpha_i \leftarrow 0$
for $j = 1 \ldots k$ **do**
  Set $\mathcal{H}_j$ as the halfplane supporting U_x with respect to c_j in the Voronoi diagram.
Call **SamplePolytope**$(\{\mathcal{H}_1, \ldots, \mathcal{H}_k\}, m)$ to generate m samples $z_1, z_2, \ldots z_m \in U_x = \cap \mathcal{H}_j$.
for $i = 1 \ldots m$ **do**
  Compute $s = \arg\min_{j=1\ldots k} d(z_i, c_j)$.
  $\alpha_s = \alpha_s + 1/m$
Return $(\alpha_1, \ldots, \alpha_k)$.

$\mathcal{H}$ consisting of all vectors orthogonal to $\mathcal{H}$. Thus, we can project all points onto $\mathcal{H}$ while retaining the same volume ratios as in the original space. This effectively reduces the problem to a k-dimensional space. The actual projection is performed by doing an singular value decomposition on the $k \times d$ matrix of the cluster representatives. Once this transformation is done, we call AFFINITY as before.

The resulting algorithm computes the affinity scores for a point in time $O(k^3/\varepsilon^2)$.

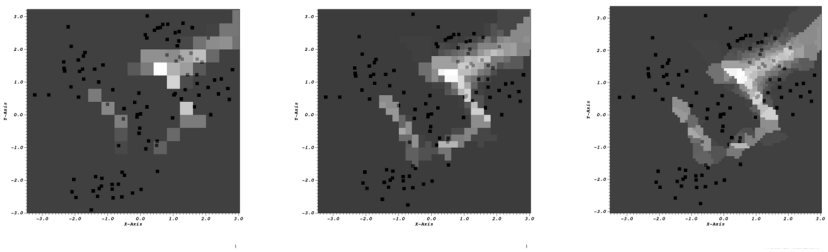
Progressive Refinement of Affinity Scores: In many applications, we care only about points with low stability since they define decision boundaries. But most points are likely to have high stability scores, and computing the scores of all points is wasteful. We describe a progressive refinement strategy that "zooms in" on the unstable points quickly. We begin with a very coarse grid on the data. For each cell, we first compute the stability score of points at the corners of the cell. If the corners are highly stable, we skip this cell, else we subdivide it further and repeat. We seed the process with a grid that has n cells (and therefore is subdivided into $n^{1/d}$ segments in each dimension.

We show the effect of this progressive refinement method for two dimensional data in Figure 8. The heatmap on the left only contains $\sqrt{n}$ cells and the one in the middle contains $10\sqrt{n}$ cells. Note that the middle heat map is very similar to the heatmap on the right that uses no refinement strategies at all, and uses far fewer stability evaluations.

VI. EXPERIMENTS

We demonstrate the benefits of affinity scores in this section. We show that

1) affinity scores identify points on the true cluster boundary which is useful to determine how a particular point affects the clustering of data.



(a) Heatmap with very coarse gridding

(b) Heatmap with moderate gridding

(c) Heatmap of stability computed on all points

Figure 8. Reducing computation through progressive refinement

2) affinity scores can be used to speed up clustering by actively selecting points that matter.
3) aggregated stability scores help with determining clusterability and model selection.
4) our method is practical and scales well with dimensionality and data size.

Data Setup: In two and three dimensions, affinity scores can be calculated via direct volume computations. We use built-in routines provided by CGAL (http://www.cgal.org) to compute the scores exactly and validate our sampling-based algorithm. For higher dimensional data, we perform the initial data transformation (if needed) in C and use a native routine for **Hit-And-Run** in MATLAB. All experiments are run on a Intel Quad Core CPU 2.66GHz machine with 4GB RAM. Reported times represent the results of averaging over 10 runs.

We created a synthetic dataset in $\mathbb{R}^2$ namely, *2D5C* for which data is drawn from 5 Gaussians to produce 5 visibly separate clusters with 100 points each. We also use a variety of datasets from the UCI repository. See Table I for details.

Dataset	#Points	#Dimensions	#Clusters
Soybean	47	35	4
Iris	150	4	3
Wine	178	13	3
MNIST (Training)	10000	784	10
Protein	17766	357	3
Adult	32561	123	2
MNIST (Test)	60000	784	10
CodRNA	488565	8	2
Covtype	581012	54	7

Table I
DATASETS.

A. Using Affinity Scores to Identify Poorly Clustered Points

We start by evaluating how well affinity scores in general (and stability specifically) pick out points that are "well assigned" or "poorly assigned". The MNIST digits data set is a good test case because it contains ground truth (the actual labeling) and we can visually inspect the results to see how the method performed.

We run a k-means algorithm on the MNIST test data and compute affinity scores of the points. We sort each digit cluster by the stability score and then pick one element at random from the top 10 and one from the bottom 10. Figure 9 shows the results for four digits The first row shows points that had high stability in the clustering (close to 1.0 in each case). We can see that the digits are unambiguous. The second row shows digits from the unstable region (the top affinity scores are 0.38, 0.46, 0.34 and 0.42 respectively). Notice that in this case the digits are far more blurred. In fact, the 4 and 9 look similar, as do the 0 and 6. The second highest affinity scores for the ones in the bottom row are 0.21, 0.19, 0.24 and 0.28 and they correspond to clusters {4, 0, 9 and 7}.

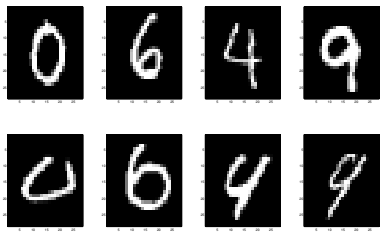


Figure 9. Results of running k-means on MNIST training data. First row: high affinity. (L-R) 0.96, 1.0, 1.0, 0.92. Second row: low affinity: (L-R) 0.38, 0.46, 0.34, 0.42.

We also validate the affinity scores against the results produced by probabilistic modeling. We run an EM algorithm to estimate the data parameters for a Gaussian mixture model and use the final cluster centers obtained to run our volume-stealing based stability method. To get a holistic view of the label affinities (instead of just looking at the maximum affinity value), we compute the entropy of the affinity score for each point (note that the affinity scores sum to 1 for each point), and we also compute the entropy of the conditional

probabilities obtained from the EM algorithm for each point. We now have two vectors of entropies, and we measure their correlation using Pearson's linear correlation coefficient. For 2D5C, Soybean and the Iris data sets, we obtain a correlation of 0.922, 0.893 and 0.935 respectively.

This further shows that affinity scores capture the strength of assignment of a point to a cluster. We reiterate that our approach merely requires the user to present a clustering obtained by *any* algorithm.

B. Using Affinity Scores to Accelerate Clustering

Most clustering algorithms take time that is non-linear in the number of points. Intuitively, points at the core of a cluster are less useful in determining the cluster boundaries, but there are more of them. Ideally, we'd like to subsample points in the core, and supersample points on the boundary to get a subset of points that can effectively recover the true clustering. Since many clustering algorithms run in time quadratic in the number of points, a good heuristic to obtain fast algorithm is to try and sample $O(\sqrt{n})$ such "good points".

We will use stability scores to identify these points in two-stage iterative approach. Firstly, we run a k-means++[28] seeding step to initialize k cluster centers. We then compute stability scores for all points and set the stability threshold at $\sigma(\mathbf{x}) = 0.5$. We fix a fraction $0 < \alpha < 1$ (set by cross validation) and then select a sample of points of size $5\alpha\sqrt{n}$ from the pool of stable points, selecting the remaining $5(1-\alpha)\sqrt{n}$ points at random from the unstable pool. In order to remove anomalies arising from any specific clustering method, we then run a spatially-aware consensus procedure[29] on this small set using k-means, hierarchical agglomerative clustering (single-linkage, average-linkage and complete-linkage variants) and DBSCAN[30] as the seed clusterings. We then assign all remaining points to their nearest cluster center. We compare this to running the same consensus procedure with all the points.

Table II summarizes the data sets used, and the sample sizes we used in each case. Figure 10 summarizes the results. In each case, the speedup over a full clustering approach is tremendous – typically a 25x speedup. Moreover, the accuracy remains unimpaired: above each bar is the Rand index comparing the clustering produced (active or full) to ground truth. In all data sets, the numbers are essentially the same, showing that our method produces as good a clustering as one that uses all the data.

As a baseline to evaluate our method, we also compared our approach with a random baseline, where we merely picked a random sample of the same size. We also measured the Rand index of the resulting clusterings, and the corresponding numbers were 0.49 for CovType, 0.55 for CodRNA, 0.81 for MNIST, and 0.48 for Protein. In all cases, our method improved over the random baseline, thus demonstrating its effectiveness at finding good clusterings.

Dataset	Points	Samples	# Stable	# Unstable
Protein	17766	665	499	166
MNIST (all)	70000	1323	992	331
CodRNA	488565	3495	2621	874
Covtype	581012	3810	2858	952

Table II
DATA SETUP FOR ACTIVE CLUSTERING.

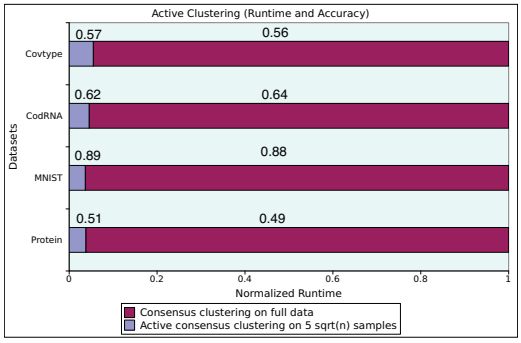


Figure 10. Performance of active sampling for consensus clustering. Rand Index is displayed above the bar for each method and each data set

C. Using Affinity Scores for Model Selection and Clusterability

While affinity scores are local, we can compute an aggregate score for a clustering by averaging the stability scores for each point. We now show that this aggregated score acts as a measure of clusterability and has useful properties that make it more effective in model selection.

(a) Choosing k. Determining the correct number of clusters for a given data is a difficult problem in clustering, especially in an unsupervised setting. The standard approach is to use some variant of the "elbow method" to analyze the trade-off curve between number of clusters and clustering cost. Since splitting a cluster typically improves the clustering cost, these methods attempt to find locations where the gradient changes dramatically, or where a point of "diminishing returns" is reached in further splitting.

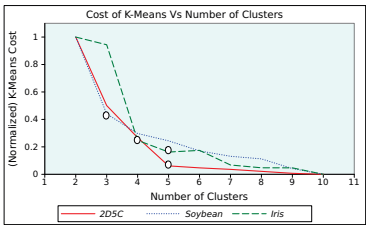


Figure 11. k-means cost Vs Number of clusters

Aggregate stability is more sensitive to splits of "good clusters". When we split a good cluster we actually *decrease* the average stability of the clustering, because all points along the boundary of the new cluster used to be very stable and now will no longer be so.

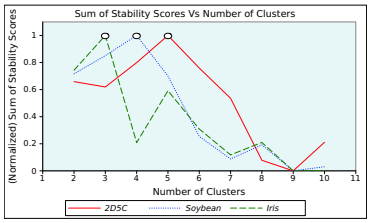


Figure 12. Average stability cost Vs Number of clusters

We demonstrate this behavior by plotting the cluster cost and average stability score for a variety of data sets from Table I. The k-means algorithm cost is plotted in Figure 11 and the average stability is plotted in Figure 12. We see that for each data set, the maximum stability is achieved at precisely the number of clusters prescribed by ground truth. In contrast, the k-means cost function strictly decreases, and it is more difficult to identify clear "elbows" at the right number of clusters.

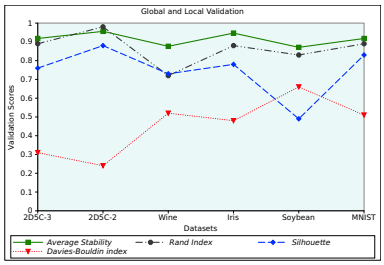


Figure 13. Aggregate Stability Vs Global Stability.

We also compare aggregate stability to standard measures of global stability like the silhouette method, the Rand index, and the Davies-Bouldin index[31]. As we can see in Figure 13, all measures behave consistently on the data sets (note that the Davies-Bouldin index is *smaller* when the clustering is better). This shows that aggregate stability acts like a global quality measure while still retaining local structure.

(b) Data Clusterability. Another use for aggregate stability is as measure of clusterability. We illustrate this by computing the aggregate stability for a clustering of five Gaussians with varying (but isotropic) covariance for each cluster. Figure 16 shows what the different clusterings look like. As we can see, the data becomes progressively less clustered as the variance increases, and therefore becomes less "clusterable".

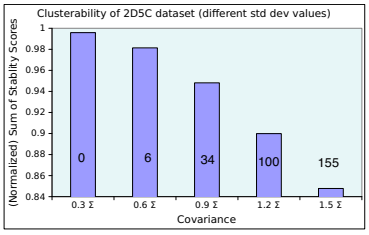


Figure 14. Clusterability of *2D5C* data: Average stability scores dip as variance increases.

Figure 14 illustrates the aggregate stability scores for these clusterings: as we can see, the scores drop similarly, and by the time we reach the fifth instance (which is essentially unclusterable), the stability numbers have dropped to nearly zero. We also annotate the graphs with the number of unstable points (with threshold $\sigma(\mathbf{x}) = 0.5$) to illustrate that the average stability is reducing consistently.

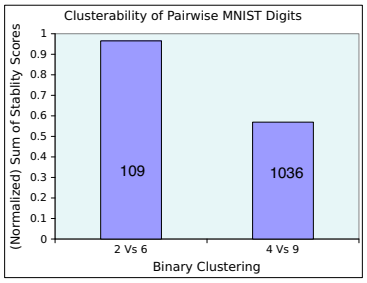


Figure 15. Clusterability of two different pair of digits in the MNIST data

As another illustration of this, we plot in Figure 15 the aggregate stability of two different pairs of numbers in the MNIST dataset (2 vs 6) and (4 vs 9). As we have seen earlier, the $2-6$ set is easier to distinguish than the $4-9$ set, and this is reflected in the different stability scores for the clustering on these two pairs.

D. Evaluating Performance

Finally, we present an evaluation of the performance of our method in terms of accuracy and running time. To validate the quality of the results, we can compare our sampling-based method to the exact scores we can obtain in two and three dimensions as described earlier. Table III illustrates this for the 2D5C and 3D5C data sets. We note that these error reports come from choosing 1000 samples after a burn-in of 1000 samples (this corresponds to an error $\varepsilon = 0.04$). As we can see, the reported error is well within the predicted range.

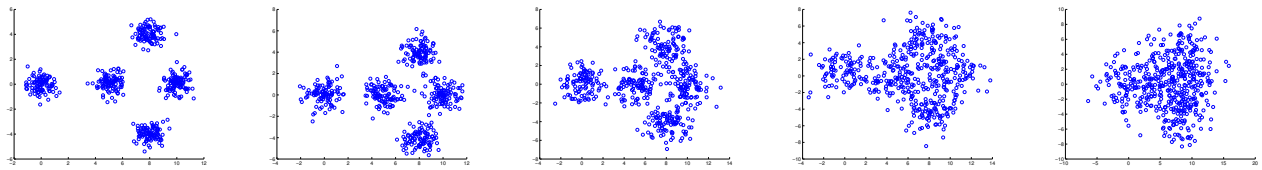
Table III also presents running times for the affinity score computation. We note that the running times reported are the total for computing the affinity scores for *all* points. We only report the time taken by the sampler; the preprocessing affine transformation is dominated by the sampling time. In all cases, we used 1000 samples to generate the estimates. Note that the procedure is extremely fast, even for the very high dimensional MNIST data.

Dataset	n	d	k	Time (sec)	Error
2D5C	500	2	5	0.11 ± 0.005	$\pm$ 0.02
3D5C	500	3	5	0.19 ± 0.008	$\pm$ 0.035
IRIS	150	4	3	0.24 ± 0.012	-
Soybean	47	35	4	0.31 ± 0.08	-
MNIST (test)	10000	784	10	0.58 ± 0.5	-

Table III
RUNTIMES AND EMPIRICAL APPROXIMATION TO EXACT AFFINITY

ACKNOWLEDGEMENTS

We are thankful to Jeff M. Phillips for discussions on local notions of stability and in particular for his suggestions on



(a) Very low (b) Low (c) Moderate (d) High (e) Very High

Figure 16. Five Gaussians with varying variance

reducing dimensionality to compute the affinity scores and for point out the correct sample complexity for ε-samples.

REFERENCES

[1] R. Xu and D. Wunsch, *Clustering*. Wiley-IEEE Press, 2009.

[2] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *J. Intell. Inf. Syst.*, vol. 17, no. 2-3, pp. 107–145, 2001.

[3] M. Houle, H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Can shared-neighbor distances defeat the curse of dimensionality?" in *SSDBM*. Springer, 2010, pp. 482–500.

[4] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu, "Understanding of internal clustering validation measures," in *Proceedings of the 2010 IEEE International Conference on Data Mining*, ser. ICDM '10, 2010, pp. 911–916.

[5] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, Nov. 1987.

[6] C. A. Sugar and G. M. James, "Finding the number of clusters in a dataset: An information-theoretic approach," *Journal of the American Statistical Association*, vol. 98, no. 463, pp. pp. 750–763, 2003.

[7] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001.

[8] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012.

[9] T. Hofmann and J. M. Buhmann, "Active data clustering," *Advances in Neural Information Processing Systems*, pp. 528–534, 1998.

[10] B. Eriksson, G. Dasarathy, A. Singh, and R. Nowak, "Active clustering: Robust and efficient hierarchical clustering using adaptively selected similarities," *arXiv preprint arXiv:1102.3887*, 2011.

[11] S. Ben-David, U. von Luxburg, and D. Pál, "A sober look at clustering stability," in *COLT*, 2006, pp. 5–19.

[12] A. Ben-Hur, A. Elisseeff, and I. Guyon, "A stability based method for discovering structure in clustered data," in *Pacific Symposium on Biocomputing*, 2002, pp. 6–17.

[13] J. Bezdek and N. Pal, "Some new indexes of cluster validity," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 28, no. 3, pp. 301 –315, jun 1998.

[14] A. Elisseeff, T. Evgeniou, and M. Pontil, "Stability of randomized learning algorithms," *Journal of Machine Learning Research*, vol. 6, no. 1, p. 55, 2006.

[15] U. Von Luxburg, *Clustering Stability*. Now Publishers Inc, 2010, vol. 3, no. 3.

[16] M. De Berg, O. Cheong, M. Van Kreveld, and M. Overmars, *Computational geometry: algorithms and applications*. Springer, 2008.

[17] F. Aurenhammer, "Power diagrams: properties, algorithms and applications," *SIAM Journal on Computing*, vol. 16, no. 1, pp. 78–96, 1987.

[18] R. Sibson, *A Brief Description of Natural Neighbour Interpolation*. John Wiley & Sons, 1981, vol. 21, pp. 21–36.

[19] L. M. Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *USSR computational mathematics and mathematical physics*, vol. 7, no. 3, pp. 200–217, 1967.

[20] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[21] G. T. Toussaint, "A simple linear algorithm for intersecting convex polygons," *The visual computer*, vol. 1, no. 2, pp. 118–123, 1985.

[22] B. Chazelle, "An optimal algorithm for intersecting three-dimensional convex polyhedra," *SIAM Journal on Computing*, vol. 21, no. 4, pp. 671–696, 1992.

[23] N. J. Lennes, "Theorems on the simple finite polygon and polyhedron," *American Journal of Mathematics*, vol. 33, no. 1/4, pp. 37–62, 1911.

[24] S. Har-Peled, *Geometric approximation algorithms*. Amer Mathematical Society, 2011, vol. 173.

[25] M. Dyer, A. Frieze, and R. Kannan, "A random polynomial-time algorithm for approximating the volume of convex bodies," *J. ACM*, vol. 38, no. 1, pp. 1–17, Jan. 1991. [Online]. Available: http://doi.acm.org/10.1145/102782.102783

[26] R. L. Smith, "Efficient monte carlo procedures for generating points uniformly distributed over bounded regions," *Operations Research*, vol. 32, no. 6, pp. 1296–1308, 1984.

[27] L. Lovász, "Hit-and-run mixes fast," *Mathematical Programming*, vol. 86, no. 3, pp. 443–461, 1999.

[28] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.

[29] P. Raman, J. M. Phillips, and S. Venkatasubramanian, "Spatially-aware comparison and consensus for clusterings," in *SDM*. SIAM / Omnipress, 2011, pp. 307–318.

[30] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996, pp. 226–231.

[31] S. Petrovic, "A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters."